ABSTRACT OF THE DISCLOSURE

The present invention accepts an acyclic concurrent control-flow graph (CCFG) and produces a sequential control flow graph (SCFG) that, when executed, behaves functionally like the CCFG would if it were run on concurrent hardware. An SCFG can be easily translated into a traditional sequential programming language such as C or assembly to be executed on a traditional sequential processor.

Determining the order in which CCFG nodes will be run is the first step in the process. Control edges in the CCFG constrain the order in which CCFG nodes must run; communication between threads generally impose further constraints. An easy way to further constrain a valid order of CCFG nodes is to augment the CCFG with data dependence edges (representing inter-thread communication) and to then topologically sort the nodes in the augmented graph to produce an ordering.

Once the CCFG nodes are ordered, the procedure for producing the SCFG from the scheduled acyclic CCFG simulates the execution of the CCFG under an operating system supporting concurrent threads and creates an SCFG that, when executed, will reproduce the functional behavior of the CCFG running under this simulated operating system. The effects of context switching are largely compiled away by this simulation process. Each context switch is done by a single assignment that stores the state of the thread being suspended and a single branch that restores the state of the thread being resumed.